HEVC Test Model (HM) 3.0 인코딩 구조

김일구·양희철 (삼성전자)

Ⅰ. 시작하며

최근 들어 차세대 비디오 코덱의 표준화가 ITU (International Telecommunication Union)와 ISO (International Organization for Standardization)의 협력으로 조직된 JCT-VC (Joint Collaboration Team - Video Coding)를 중심으로 진행되고 있다. JCT-VC는 H.264/AVC의 개발을 위한JVT (Joint Video Team)가 해체된 후 만들어진ITU와 ISO의 협력기구이다. 차세대 비디오 코덱의 코드네임은 High Efficiency Video Coding (HEVC)로 현재 가장 널리 사용되고 있는H.264/AVC보다 2배의 압축효율을 가지는 것을 목표로 하고 있다. CD (Committee Draft)의 완성은 2012년 1월, 표준화의 최종 완료는 2013년으로 예정되어 있다. 본 기고에서는 HEVC의 reference model로 사용되고 있는 test model인 HM의 인코딩 구조에 대해 중점적으로 소개한다. 본 기고는 JCTVC-E602 [1]를 바탕으로 작성되었다.

Ⅱ. History

HEVC의 세번째 HM인 HM-3.0는 3월 16일에서 23일간 제네바에서 열린 제 5회 JCT-VC 미팅에서 정의되었다. 이전 미팅에서는 JCT-VC는 JCTVC-B204 [2]와 JCTVC-B205 [3]에 정의된 Test Model under Consideration (TMuC)을 정의하였다. TMuC은 HM을 확정하기 전에 임시적으로 사용된 test model이다.

광저우에서 열린 3번째 JCT-VC 미팅에서 첫번째 Working Draft (WD) 문서와 함께 최초의 HEVC Test Model인 HM-1.00이 정의되었다. HM-1.0의 기술들은 TMuC에 존재하던 기술들 중, 성능이 검증된 매우 적은 기술들로만 구성되었기 때문에 인코딩, 디코딩에 필요한 복잡도가 매우 낮아졌다. HM-2.0와 HM-3.0은 HM-1.0의 최적화를 통해서 코딩 효율과 복잡도 측면에서 HM-1.0보다 모든 면에서 향상된 성능을 보인다. HM-3.0은 이전 버전과 비교해 일관된 디자인을 갖고 있으며, 뛰어난 성능을 보이는 검증된 최소의 기술들의 집합으로 구성되어 있다. 현재 HM-3.0은 두 가지의 설정에 대응할 수 있다: 고효율 (High Efficiency, HE)과 저복잡도 (Low Complexity, LC). HM-3.0에 포함된 기술들은 〈표 1〉에 정리되어 있다. 엔트로피 코더와 ALF (Adaptive Loop Filter)를 제외한 모든 기술들은 HE와 LC에 모두 적용된다. CABAC

〈표 1〉 HM-3.0에 정의된 기술들 [1]

High Efficiency Configuration	Low Complexity Configuration
Coding Unit tree structure (8x8 up to 64x64 luma samples)	
Prediction Units	
Transform unit tree structure (3 level max.)	
Transform block size of 4x4 to 32x32 samples Mode-dependent Transform for 4x4 block	
Spatial Intra Prediction (up to 34 angular directions and Planar) Adaptive Intra Smoothing Intra Chroma Prediction using Luma samples	
DCT-based interpolation filter for luma samples (1/4-sample, 8-tap)	
DCT-based interpolation filter for chroma samples (1/8-sample, 4-tap)	
Coding Unit based Skip & Prediction Unit based merging	
Advanced motion vector prediction	
Context adaptive binary arithmetic entropy coding (CABAC)	Context adaptive VLC (CAVLC)
Deblocking filter Sample Adaptive Offset	
Adaptive loop filter	X

(Context Adaptive Binary Arithmetic Coding)과 ALF의 경우 HE에만 적용되고, CAVLC (Context Adaptive Variable Length Coding)는 LC의 경우에만 적용된다. 이러한 설정에 대한 자세한 내용은 JCTVC-E700 [4]에 정의되어 있다.

Ⅲ. HM-3.0의 전체구조

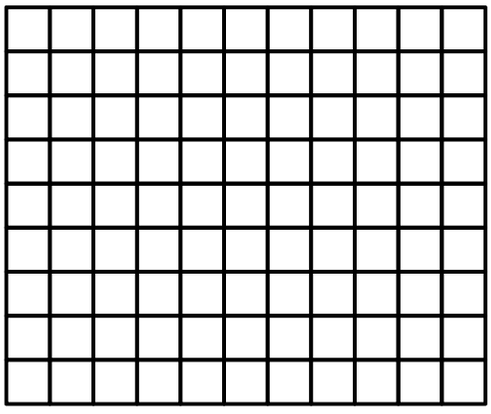
HM-3.0은 전형적인 블록기반 하이브리드 (block-based hybrid) 코덱의 구조를 가지고 있다. 기존 H.264/AVC에 사용된 기술들을 더욱 개선하거나, 그것을 바탕으로 새로운 기술들이 덧붙어 있는 형태이다. 기존 기술의 발전, 확장된 형태는 Coding Unit (CU)/Prediction Unit (PU)/Transform Unit (TU) 구조, 인트라 예측 방법, 움직임 정보 예측 방법, 공간 변환, 디블록킹 필터, CABAC, CAVLC 등이 있다. 새롭게 추가된 기술은 ALF, SAO (Sample Adaptive Offset) 등과 같은 인루프 필터링 기술들이다.

HM-3.00이 기존의 비디오 코덱과 차별화되는 점은 처리의 기본이 되는 블록이 MB (macroblock)에서 CU로 변경되었다는 점이다. 기존 MB의 경우, 고정적인 16x16 크기를 가지지만, CU의 경우, SPS (Sequence Parameter Set)에 정의된 값에 따라서 크기를 가변적으로 설정할 수 있다. 현재 HM-3.0에서는 CU의 크기는 64x64로 설정되어 있다. 가능한 가장 큰 CU는 LCU (Largest Coding Unit) 혹은 TB (treeblock)라 불린다. 본 기고에서는 일관성을 위해 LCU로 사용한다.

1. 픽쳐 분할

HM-3.0에서는 하나의 픽쳐는 겹치지 않는LCU의 연속된 형태로 분할이 된다. LCU는 하나의 2Nx2N luma 블록과 2개의 chroma 블록으로 구성되는데, 3개의 독립된 컬러 플레인으로 처리된다. LCU의 개념은 H.264/AVC의 MB와 개념적으로 동일하다. 현재 LCU의 크기는 64x64로 이것은 가장 큰 CU의 크기와 일치한다. 〈그림 1〉은 704x576크기의 픽쳐가 64x64 LCU로 나눠진 예시를 보여주고 있다.

LCU는 더 작은 크기의 CU로 분할이 가능하다. 이러한 분할의 형태는 전형적인 쿼드트리 (quadtree) 형태를 따른다. 각 CU의 분할 여부는 하나의 플래그(flag)를 통해 표현된다. 만약 분할이 된다면 4개의 동일한 크기를 갖는 Sub-CU로 분할이 된다. 분할된 작은 Sub-CU 역시 현재 CU와 동일한 형태로 재귀적으로 분할이 가능하다. 분할 가능한 CU 쿼드트리의 깊이도 역시 LCU의 크기와 더불어 SPS에 정의된다. 가장 깊은 CU 쿼드트리에 존재하는 가장 작은 크기의 CU는 SCU

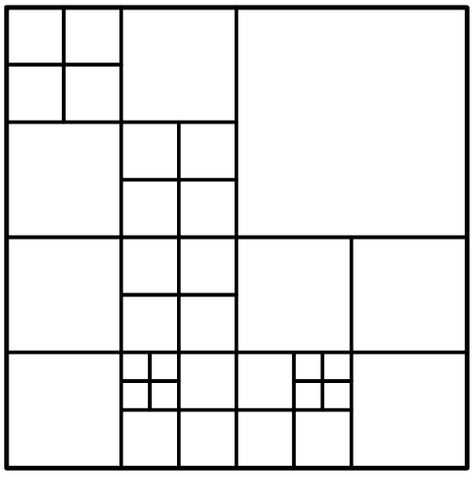


〈그림 1〉 LCU로 나눠진 픽쳐의 예시 [1]

(Samllest Coding Unit)라고 불린다. HM-3.0의 인코딩 구조는 LCU의 크기와 CU 쿼드트리의 깊이에 의해 완벽하게 정의될 수 있다. 현재 HM-3.0에서는 쿼드트리의 최대 깊이는 4로 설정되어 있고, 이 경우 SCU의 크기는 8x8이된다. 참고적으로 LCU의 크기를 16x16, CU 쿼드트리의 최대 깊이를 2로 설정할 경우H.264/AVC와 유사한 블록 구조를 가지게 된다.

2. CU 쿼드트리

HM-3.0에서 CU는 인터/인트라 예측에 이용되는 공간분할의 기본단위이다. CU는 항상 정사각형 형태를 가지면서, LCU 크기 (64x64)부터 시작하여, SCU (8x8) 크기까지 작아질 수 있다. CU는 LCU 부터 시작하여 동일크기를 갖는 4개의 Sub-CU으로 분할될 수 있다. 분할의 여부는 인코더에 의해 결정되며, 이러한 분할 과정을 통해 픽쳐의 특성에 따라서 적응적으로 다양한CU 쿼드트리 구성할 수 있다. 〈그림 2〉는 11번의 분할과정에 의해 생성된 CU 쿼드트리의 예시를 보여주고 있다. 그림에서 가장 큰 사각형은 LCU이다.

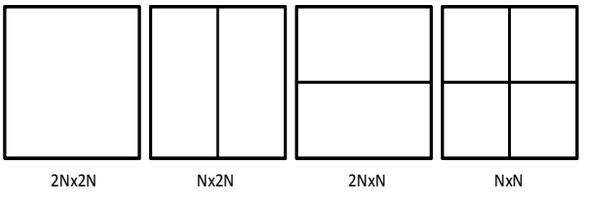


〈그림 2〉 CU 쿼드트리의 예시 [1]

3. PU의 형태

PU는 예측 프로세스과 관련된 정보를 가지고 있는 가장 기본단위이다. PU는 물체 경계를 효과적으로 표현하기 위해서 정사각형 형태로 제한되지 않는다. 각 CU는 하나 이상의 PU를 포함 할 수 있다. 〈그림 3〉은 인터CU가 가질 수 있는PU 형태를 나타내고 있다. 그림에서 2Nx2N의 경우 현재 CU와 동일하다.

인트라 CU의 경우 2Nx2N과 NxN 형태만 존재한다. 주의할 점은 모든 NxN형태의 PU는 SCU에서만 허용된다. (현재 HM-3.0의 세팅에서는 SCU의 크기는 8x8이다.)

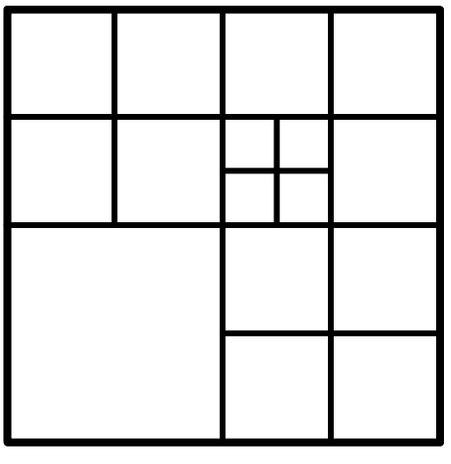


〈그림 3〉 인터 CU의 4가지 PU 형태 [1]

4. TU의 구조

TU는 공간변환과 양자화 과정에 사용되는 기본 유닛이다. TU는 항상 정사각형 형태이며 4x4에서 32x32의 크기를 갖는다. TU의 최대 크기는 HM-3.0에서 사용되는 최대 프랜스폼의 크기와 동일하다. 각 TU는 그 안에 하나 이상의 TU를 가질 수 있으면, 여러 개의 TU는 〈그림 4〉와 같이 쿼드트리 형태를 가질 수 있다. 그림에서 가장 큰 사각형은 현재 CU의 잔차 (residual) 블록이다.

TU 쿼드트리는 현재 CU에서부터 시작되며, 이에 따라서 TU 쿼드트리의 깊이는 가변적이다. 가장 깊은 쿼드트리의 깊이는 슬라이스 헤더에 정의되어 있으며, 현재 HM-3.0에서는 인터/인트라 슬라이스에서 모두 3으로 설정되어 있다. 최대 쿼드트리의 깊이값이 크면, 코딩 효율을 증가하지만, 인코딩 복잡도가 증가한다.



〈그림 4〉 TU 쿼드트리의 구조 예시 [1]

Ⅳ. 코스트 함수 (Cost Functions)

HM-3.0에서는 다양한 코스트 함수들이 모드 및 파라미터의 결정을 위해 사용된다. 본 장에서는 HM-3.0에서 사용되고 있는 에러 메트릭 (error metric) 및 코스트 함수 들에 대해 소개한다.

1. Sum of Square Error (SSE)

동일한 크기의 두 블록 사이의 차이는 다음의 식으로 구한다.

$$Diff(i,j) = BlockA(i,j) - BlockB(i,j)$$

SSE는 다음과 같은 식으로 구한다:

$$SSE = \sum_{i,j} Diff(i,j)^2$$

2. Sum of Absolute Difference (SAD)

SAD는 다음의 식을 이용해서 구한다:

$$SAD = \sum_{i,j} \left| Diff(i,j) \right|$$

3. Hadamard transformed SAD (SATD)

HM-3.0에서는 변환계수들이 양자화되어 압축되기 때문에, 일반적인 SAD보다는 하다마드 변환을 이용하면 에러를 더 정확히 예측할 수 있다. SATD는 다음과 같은 식으로 구한다:

$$SATD = (\sum_{i,j} \left| DiffT(i,j) \right|) / 2$$

위 식에서 *DiffT(i,j)*는 하다마드 변환된 두 블록간의 차이를 의미한다. 참고적으로 현재 full-pel ME (Motion Estimation)

에서는 SAD가 사용되고, sub-pel ME에서는 SATD가 사용되고 있다.

4. RD 코스트 함수들

가. 라그랑지안 계수 (Lagrangian constant)

현재 HM-3.0 인코더에서는 코스트 계산에 이용되는 람다 값은 다음과 같이 정의되어 있다.

$$\lambda_{\text{mode}} = \alpha * W_k * 2^{((QP-12)/3.0)}$$

$$\lambda_{motion} = \sqrt{\lambda_{\text{mode}}}$$

$$\alpha = \begin{cases} 1.0 - Clip3(0.0, 0.5, 0.05 * number\_of\_B\_frames) \\ 1.0 \end{cases}$$

W_k 는 인코딩 설정 및 GOP안의 계층적 레벨에 의존적으로 결정되는 가중 팩터(weighting factor) 이다. W_k 는 다음의 테이블에 의해 결정된다. SATD가 사용된 경우에는 〈표 2〉를 이용해 구해진 W_k 에 0.95를 곱해서 사용한다.

나. 예측 파라미터 결정에 사용되는 SAD 기반 코스트 함수

예측 파라미터 결정에 사용되는 코스트 $J_{pred,SAD}$ 는 다음과 같은 식으로 정의된다.

$$J_{pred,SAD} = SAD + \lambda_{pred} * B_{pred}$$

위 식에서 B_{pred} 는 현재 예측 파라미터를 이용해 코딩된 비트로 정의된다. λ_{pred} 와 SAD는 이전 장에서 각각 설명되어 졌다.

다. 움직임 파라미터 결정에 사용되는 SATD 기반 코스트 함수

움직임 파라미터 결정에 사용되는 코스트 $J_{pred,SATD}$ 는 다음과 같은 식으로 정의된다.

$$J_{pred,SATD} = SATD + \lambda_{pred} * B_{pred}$$

위 식에서Bpred 는 현재 움직임 파라미터를 이용해 코딩된 비트로 정의된다.λ pred 와SATD는 각각 위에 장에서 정의되었다.

라. 모드 결정에 사용되는 코스트 함수

모드 결정에 사용되는 코스트 함수Jmode 는 다음과 같은 식으로 정의된다.

$$J_{mode} = SSE + \lambda_{mode} * B_{mode}$$

위 식에서 B_{mode} 는 모드 결정에 필요한 비트량으로 구해진다. λ_{mode} 와 SSE는 위 장에서 각각 정의 되었다.

V. HM-3.0의 시간 예측 구조

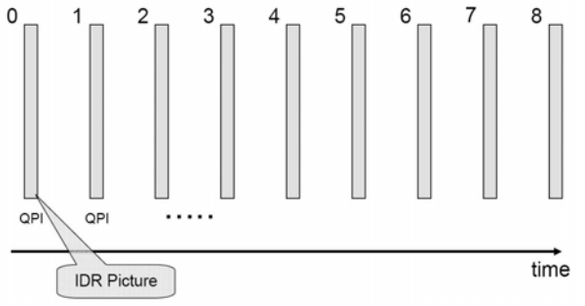
HM-3.0 인코더는 JCT-VC Common Test Condition [4]에 정의된3가지 서로 다른 형태의 시간 예측 구조를 갖는다. 참조 픽처 리스트 관리 (reference picture list management)는 각 시간 예측 구조에 따라야 한다. 현재 참조 픽처 리스트 관리는 완벽히 구현되어 있지 않다.

1. 인트라 설정 (intra only)

인트라 설정에서는, 각 픽쳐는 IDR (Instantaneous Decoding Refresh) 픽쳐로 코딩되야 한다. 모든 픽쳐는 시간적 예측을 사용하지 않는다. 비디오 전체 혹은 픽쳐 내부에서QP의 변화는 허용되지 않는다. 〈그림 5〉가 인트라 설정을 보여주고 있다. 각 픽쳐 위의 숫자는 인코딩 순서를 나타내고 있다.

〈표 2〉 W_k 의 결정 방법 [1]

k	Hierarchy level	Use RDO-Q	Slice type	Number of B-frames	Referenced	W_k
0	0	1	I	–	–	0.57
1	0	–	I or GPB	〉0	–	0.68
2	0	–	I or GPB	0	–	0.85
3	〉0	–	GPB	–	–	0.68
4	〉0	–	B	–	0	0.68 * Clip3(2.0, 4.0, (QP-12)/6.0)
5	〉0	–	B	–	1	0.68 * Clip3(2.0, 4.0, (QP-12)/6.0) * 0.80



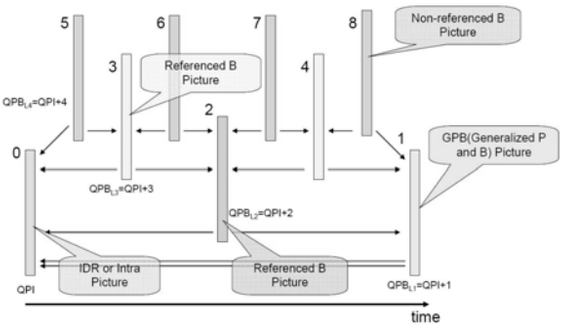
〈그림 5〉 인트라 설정 [1]

2. 저지연 설정 (Low-Delay)

두 종류의 저지연 설정이 저지연 상황에서 성능 평가를 위해 정의되어 있다. 저지연 설정에서는 전체 비디오 중에서 오직 첫 픽쳐만이 IDR 픽쳐로 코딩된다. 의무적인 저지연 테스트에서는 IDR 픽쳐를 제외한 모든 픽쳐들은 Generalized P and B-picture (GPB) 픽쳐로 코딩된다. GPB 픽쳐들은 자신의 POC (Picture Order Count) 보다 작은 POC를 갖는 참조 픽쳐들만 참조할 수 있다. (예를 들면 RefPicList0와 RefPicList1에 포함된 모든 참조 픽쳐들은 현재 픽쳐보다 출력 순서상 시간적으로 이전에 있어야 한다.). RefPicList0와 RefPicList1 의 내용은 동일해야 하며 두 리스트는 sliding-window management process를 이용하여 갱신된다. 〈그림 6〉이 의무적인 저지연 설정을 묘사하고 있다. 각 픽쳐위의 숫자들은 인코딩 순서이다. 인터 코딩된 픽쳐들의 QP는 시간적인 계층구조에 따라서 IDR 픽쳐의 QP에 offset을 더한 형태로 결정된다. 추가적인 선택적 저지연 설정에서는 모든 인터 픽쳐들은 하나의 참조 픽쳐 리스트 (RefPicList0)를 사용하는 P 픽쳐로 코딩된다.

3. 임의 접근 설정 (Random Access)

임의 접근 설정에서는 Hierarchical B 구조가 사용된다. 〈그림 7〉이 임의 접근 설정을 묘사하고 있다. 각 픽쳐위의 숫자들은 인코딩 순서이다. 임의 접근 설정에서는 인트라 픽쳐들이 1초 간격으로 포함되야 한다. 전체 비디오 중에서 첫 픽쳐는 IDR로 코딩 (0번 픽쳐)되고 나머지 인트라 픽쳐들은 non-IDR로 코딩된다. ("Open GOP"). 출력 순서상 인트라 픽쳐들 사이에 위치한 픽쳐들은 B 픽쳐로 코딩되어야 한다. 가장 낮은 계층의 픽쳐들은 I 픽쳐와 GPB 픽쳐를 참조할 수 있는 GPB 픽쳐로 코딩되어야 한다 (0번 1번 픽쳐). 두번째와 세번째 계층은 참조가 가능한 B 픽쳐로 코딩 (2, 3, 4번 픽쳐)되고 마지막 계층의 픽쳐는 참조하지 않는 B 픽쳐로 코딩된다 (5, 6, 7, 8번 픽쳐). 각 인터 코딩된 픽쳐들의 QP는 시간 계층에 따라서 인트라 코딩된 픽쳐의 QP에 offset을 더한 형태로 결정된다.

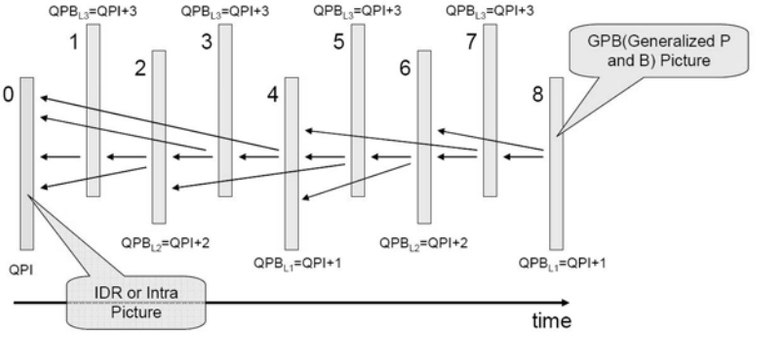


〈그림 7〉 임의 접근 설정 [1]

Ⅵ. 인코딩 파라미터의 결정

1. CU구조의 결정 방법

HM-3.0에서는 CU 단위로 모든 프로세스를 진행한다. (주: fine granular slice의 경우 마지막 노드의 CU에서 슬라이스가 나눠질 수 있으나, 여기서는 설명의 편의를 위해, LCU 단위 프로세싱을 가정한다.) CU가 다음 레벨로 나눠질지 아닐지는 현재 레벨에서 프로세싱을 진행하는 경우의 코스트와 4



〈그림 6〉 의무적인 저지연 설정 (GPB) [1]

개로 나눠진 Sub-CU들의 코스트의 합을 비교하여 코스트 값이 작은 경우를 선택한다. 현재 레벨을 d라고 하고 다음 레벨은 d+1이 되고 다음과 같은 수식으로 나타낼 수 있다.

$$J_{mode,\ d} = SSE_d + \lambda_{mode} * B_{mode,\ d}$$
$$J_{mode,\ d+1} = \Sigma SSE_{d+1} + \lambda_{mode} * \Sigma B_{mode,\ d+1}$$

$J_{mode,\ d}$ 가 $J_{mode,\ d+1}$ 다 더 작은 경우 분할이 되지 않고 그렇지 않은 경우는 d+1 레벨로 분할이 된다. d+1로 분할이 결정된 경우에는 분할된 4개의 CU에 대해서 다시 d+2 레벨의 코스트와 비교해서 분할 여부를 다시 결정한다.

2. 인트라 예측 모드 및 파라미터 결정

HM-3.0의 인트라 기술은 PU의 크기에 따라서 최대 34의 방향과 플라나 (Planar) 예측 모드를 제공한다. PU 의 크기가 4x4, 8x8, 16x16, 32x32, 64x64에 대해서, 각각 17, 34, 34, 34, 3 개의 방향이 사용된다. luma에 대한 최적의 인트라 예측 모드는 다음과 같은 방법으로 결정된다.

가장 먼저 간략화된 모드 결정과정이 수행된다. 예측 코스트$J_{pred,SATD}$ 가 모든 예측 모드에 대해 계산되고 그 중에서 작은 코스트를 갖는 순서대로 미리 결정된 개수 만큼의 예측 모드들이 선택된다. (4x4-8x8 PU에 대해서는 8개, 16x16-64x64 PU에 대해서는 3개). 이러한 간략화된 예측 과정에서 인트라 모드를 위한 비트는B_{pred} 로 나타낸다. 그 후, 줄어든 예측 모드와 MostProbableMode에 대해서 RD 코스트 J_{mode} 를 구해서 최소의 코스트를 갖는 예측 모드를 선택한다. RD 결정 과정에서 예측 파라미터들과 변환 계수들은 B_{mode} 를 계산하는데 사용된다. chroma 예측 모드의 결정은 모든 가능한 chroma 예측 모드에 대해서 RD 코스트를 계산한 후 최소의 코스트를 갖는 예측 모드를 선택한다. chroma 역시 RD 결정 과정에서 예측 파라미터들과 변환 계수들은 B_{mode} 를 계산하는데 사용된다 .

3. 인터 예측 모드 및 파라미터의 결정

가. 움직임 파라미터의 예측

HM-3.0 인코더에서는 인터 CU는 여러 개의 인터 PU로 나눠질 수 있다. 각각의 PU들은 하나 이상의 움직임 벡터들 (참조 픽처 리스트에서 하나씩), 움직임 벡터들에 대응되는 참조 픽쳐 인텍스 (ref_idx_lX, X는 0 혹은 1), 움직임 예측 방향 (inter_pred_flag)으로 구성된 한 세트의 움직임 파라미터들의 집합을 가질 수 있다. 하나의 인터 CU는 다음의 모드 중 하나의 모드로 코딩된다: MODE_SKIP, MODE_INTER. MODE_SKIP의 경우, 더 작은 PU로 나눠지지 않으며 움직임 파라미터들이 CU 자체에 할당되며, PU의 크기는 PART_ 2Nx2N가 된다. 반대로 MODE_INTER의 경우 최대 4가지의 형태의 PU로 나눠질 수 있다. 예측 모드와 PU의 형태는CU 레벨에서 "part_type"을 이용해서 디코더에 전송된다. SCU보다 큰 크기를 갖는 CU의 MODE_INTER는 세가지 형태의 PU를 가질 수 있다. (PART_2Nx2N, PART_2NxN and PART_Nx2N). PART_NxN는 SCU 크기를 갖는 CU만 선택할 수 있다. 각 PU는 PU기반 merge 모드 혹은 움직임 예측을 이용한 인터 모드 중 하나를 선택할 수 있다. 다음 장에서는 어떻게 luma움직임 파라미터를 결정하는지 기술한다. chroma의 경우 대응되는 luma CU의 움직임 정보를 이용해서 계산할 수 있다. luma와 동일한 참조 픽쳐 인덱스와 움직임 예측 방향을 사용한다.

나. 움직임 벡터 예측

각 PU에 대해서 최적의 움직임 벡터 예측치를 다음의 방법으로 구한다. 먼저 RefPicListX에 대해서 주변의 움직임 파라미터 정보를 이용해서 한 세트의 움직임 벡터 예측치의 후보들을 끌어낸다. 후보 집합들 중에서 최적의 움직임 파라미터를 코스트 $J_{pred,SAD}$를 최소화하는 움직임 파라미터로 선택한다.

움직임 벡터 예측 (Motion Estimation) 에서는 주어진 탐색 범위 안에서 최소의 코스트를 갖는 intege-pel의 위치를 먼저 찾는다. 현재 HM-3.0에서는 다이아몬드 탐색 방법을 이용해서 integer-pel의 위치를 탐색한다. 더 빠른 움직임 벡터 예측을 위해 EPZS 등과 같은 개선된 방법이 고려되고 있다. integer-pel의 위치가 선택되면, 주위의 8개의 half-pel 위치의 코스트를 계산하여 최소 코스트를 갖는 half-pel 위치를 찾는다. 그 후에 다시 8개의 주위 quarter-pel 위치의 코스트를 계산 한 후 최소의 코스트를 갖는 위치를 최적의 움직임 벡터(MV)로 결정한다.

다. MODE_SKIP

MODE_SKIP의 경우, 움직임 파라미터는 merge 모드를 이용해서 구해진다. 이 경우, 최적의 움직임 파라미터는 merge 후보들 중에서 최소의 코스트 J_{mode} 를 갖는 후보로 결정한다. 코스트 계산에 이용되는 비트B_{mode} 는skip_flag 및 merge_idx를 고려해서 구해진다. MODE_SKIP에서는 잔차신호가 전송이 되지 않기 때문에, SSE를 이용해서 코스트를 구한다.

라. MODE_INTER

CU가 MODE_INTER로 코딩될때, 움직임 파라미터 결정은

먼저ME cost $J_{pred,SATD}$ 를 이용한다. PU의 형태가 “PART_2Nx2N” 혹은 “PART_NxN”의 경우, 모든 PU들은 merge 모드 혹은 인터 모드로 코딩될 수 있다. 그러나PU 형태가 “PART_2NxN” 혹은 “PART_Nx2N”일 경우에는, 첫번째 PU는 항상 merge 모드를 사용해야 한다. 그러나 두번째 PU는 merge 모드 혹은 인터 모드 둘 중에 하나를 사용해서 코딩될 수 있다. 참고적으로 이러한 기술을 Inferred merge라고 부른다.

merge 모드의 경우, 움직임 파라미터의 결정 과정은 merge 모드에서 사용될 움직임 벡터의 후보집합을 선정한다. 만약 유효한 merge 후보가 없다면, HM-3.0 인코더는 zero 움직임 벡터를 후보로 넣는다. 그렇지 한다면 (하나 이상의 merge 후보가 존재하는 경우), 모든 merge 후보들에 대해서 ME cost $J_{pred,SATD}$를 계산하고, 최소의 ME cost를 갖는 merge 후보를 현재 PU의 움직임 파라미터로 결정한다. 이 경우 원본 영상과 예측 영상 사이의 SATD를 에러 메트릭으로 사용하고 merge_idx를 고려한 비트를 B_{pred}로 이용한다.

인터 모드의 경우, 최적의 움직임 파라미터는 움직임 예측 프로세스를 이용해서 구해진다. 움직임 예측 프로세스에서는, 최적의 움직임 파라미터는 코스트 $J_{pred,SATD}$,를 최소화하는 움직임 파라미터를 선택한다. Merge모드와 동일하게 원본 영상과 예측 영상 사이의 SATD를 에러 메트릭으로 사용하고 inter_pred_flag, ref_idx_lX, mvd_lX, mvp_idx_lX (X는 0 혹은 1) 를 고려한 비트를 B_{pred}로 이용한다.

Merge 모드와 인터모드의 최적의 움직임 파라미터가 결정된 후에, 두개의 코스트를 다시 계산하여 더 작은 코스트를 갖는 모드를 현재 PU의 모드로 결정한다.

4. Intra/inter/PCM 모드의 결정

인터 모드 CU에 대해서 아래의 모드 결정 과정이 적용된다.

1. MODE_SKIP과 PART_2Nx2N 크기를 갖는 MODE_INTER를 merge 모드를 이용해서 구한 코딩 코스트(J_{mode})를 각각 구한 후 더 작은 코스트를 최소 CU 코딩 코스트 J 에 저장한다.
2. PART_2Nx2N를 갖는 MODE_INTER를 코딩하여 코스트(J_{mode})를 구한 후에, *J_{mode}가 J보다 작을 경우 J를 J_{mode}* 값으로 세팅한다.
3. 현재 CU의 크기가 SCU와 동일하면 4번으로 간다. 그렇지 않으면 5번으로 간다.
4. PART_NxN 크기를 갖는 MODE_INTER를 코딩하여 코스트(J_{mode})를 구한 후에, *J_{mode}가 J보다 작을 경우 J를 J_{mode}* 값으로 세팅한다.
5. PART_2NxN 크기를 갖는 MODE_INTER를 코딩하여 코스트(J_{mode})를 구한 후에, *J_{mode}가 J보다 작을 경우 J를 J_{mode}* 값으로 세팅한다. 이 경우, CU 크기가 SCU 크기와 동일하지 않다면, 첫번째 PU는 merge 모드를 이용해서 코딩해야 한다.
6. PART_Nx2N 크기를 갖는 MODE_INTER를 코딩하여 코스트(J_{mode})를 구한 후에, *J_{mode}가 J보다 작을 경우 J를 J_{mode}* 값으로 세팅한다. 이 경우, CU 크기가 SCU 크기와 동일하지 않다면, 첫번째 PU는 merge 모드를 이용해서 코딩해야 한다.
7. PART_2Nx2N 의 크기를 갖는 MODE_INTRA 를 코딩하여 코스트(J_{mode})를 구한 후에, *J_{mode}가 J보다 작을 경우 J를 J_{mode}* 값으로 세팅한다.
8. 현재 CU의 크기가 SCU와 동일하면 9번으로 간다. 그렇지 않으면 10번으로 간다.
9. 현재 CU의 크기가 최소 TU 크기보다 큰 경우에는 PART_NxN 크기를 갖는 MODE_INTRA를 코딩한 후 코스트(J_{mode})를 구한 후에, *J_{mode}가 J보다 작을 경우 J를 J_{mode}* 값으로 세팅한다.
10. 현재 CU의 크기가 최소 PCM 모드 크기보다 크거나 동일한지 체크한다. 크거나 동일하다면 11번으로 이동하고 그렇지 않다면 12번으로 이동한다.
11. 다음의 조건을 만족하면, 코스트(*mode*)를 구한 후에 *J_{mode}가 J보다 작을 경우 J를 J_{mode}* 값으로 세팅한다.
    - 현재까지의 코스트 J 가 인풋 이미지 블록의 PCM 샘플 데이터의 코스트보다 큰 경우
    - 현재까지의 코스트 J 인풋 이미지 블록의 PCM 샘플 데이터의 코스트에 λ_{mode} 를 곱한 값 보다 큰 경우
12. 비트 코스트 B_{mode} 를 CU split 비트를 고려해서 다시 구하고, 최소 코스트 J를 다시 계산한다.

PCM 모드를 제외한 J_{mode} 를 계산할 때는, 원본 데이터에서 인트라/인터 예측 데이터를 뺀 후에 잔차신호를 얻고, 공간 변환과 양자화를 적용해서 TU 쿼드트리를 만들어 잔차신호를 코딩한다. 코딩에 필요한 정보들 (skip_flag, merge_flag, merge_idx, pred_type, pcm_flag, inter_pred_flag, reference picture indices, motion vector(s), mvp_idx, intra prediction mode signaling), residual coded data의 비트 정보는B_{mode}에 포함된다. 디코딩된 정보를 이용해서 *SSE* 를 구한다. MODE_SKIP의 경우 예측 정보를 디코딩된 정보로 이용한다.

PCM 모드의 경우, J_{mode}, 코딩에 필요한 정보들 (skip_flag, pred_type, pcm_flag, pcm_alignment_zero_bit), PCM sample data의 비트들이B_{mode}에 포함된다. *SSE* 값은0으로 세팅된다.

CU 레벨 모드의 결정은 각 CU 깊이 별로 재귀적으로 결정되고 최종적인 CU 쿼드트리의 구조는 최종적으로 LCU 레벨에서 결정된다.

Ⅶ. 인루프필터: 디블록킹 필터, SAO, ALF

디블록킹 필터, SAO, ALF의 인코딩 파라미터의 결정 방법의 경우 HM-4.0에서 많은 변화가 있을 것으로 예상되어 본 기고문에서는 제외하였다. 관심있는 분들은 JCT-VC 사이트를 참조하여 확인할 수 있다 [5].

Ⅷ. 나가면서

본 기고에서는 현재 관심이 고조되고 있는 HEVC의 test model로 사용되고 있는 HM-3.0의 인코딩 기술에 대해 살펴보았다. 현재 표준화가 활발히 진행되고 있는 단계이기 때문에, 차후에 관련 기술들이 변경될 수 있음을 밝힌다. 변경된 사항은 JCT-VC 사이트를 참조하여 확인할 수 있다 [5]. 참고적으로 2011년 8월 현재 HM-4.0이 정의되어있다.

참고문헌

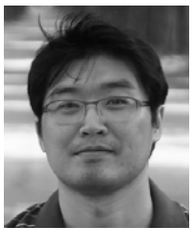
[1] JCT-VC, "HM3: High Efficiency Video Coding (HEVC) Test Model 3 Encoder Description", JCTVC-E602, 5th JCT-VC Meeting: Geneva, CH, 16-23 March, 2011.

[2] JCT-VC, "Encoder-side description of Test Model under Consideration", JCTVC-B204, 2nd Meeting: Geneva, CH, 21-28 July, 2010.

[3] JCT-VC, "Test Model under Consideration", JCTVC-B205, 2nd Meeting: Geneva, CH, 21-28 July, 2010

[4] JCT-VC, "Common test conditions and software reference configurations", JCTVC-E700, March, 2011.

[5] JCT-VC document management system http://phenix.int-evry.fr/jct/



김 일 구

1999년 서울대학교 전기공학부 학사.
2001년 서울대학교 전기공학부 석사.
2006년 서울대학교 전기공학부 박사.
2006년~현재 삼성전자 책임 연구원.
〈관심분야〉 image/video compression, processing and standardization

양 희 철

2009년 광운대학교 전자공학과 학사.
2011년 성균관대학교 정보통신공학부 석사.
2011년~현재 삼성전자 책임 연구원.
〈관심분야〉 image/video compression, processing and standardization